



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных
технологий

Отчет по практической работе №4

по дисциплине «Технологии разработки программных приложений»

Тема практической работы: «Docker»

Выполнил:

Студент группы ИКБО-42-23

Голев С.С.

Проверил:

Доцент кафедры МОСИТ,
кандидат технических наук, доцент
Жматов Д.В.

СОДЕРЖАНИЕ

ХОД РАБОТЫ	3
1 Образы	3
2 Изоляция.....	3
3 Работа с портами	4
4 Именованные контейнеры, остановка и удаление	5
5 Постоянное хранение данных	5
7 Dockerfile	6
ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ	8
ВЫВОД.....	10

ХОД РАБОТЫ

1 Образы

Загрузим последний образ Ubuntu и образ конкретной версии.

```
PS C:\Users\semen> docker pull ubuntu:12.04
12.04: Pulling from library/ubuntu
83251ac64627: Download complete
d62ecaceda39: Download complete
6d93b41cfc6b: Download complete
d8868e50ac4c: Download complete
589bba2f1b36: Download complete
Digest: sha256:18305429afa14ea462f810146ba44d4363ae76e4c8dfc38288cf73aa07485005
Status: Downloaded newer image for ubuntu:12.04
docker.io/library/ubuntu:12.04
PS C:\Users\semen> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu	latest	1e622c5f073b	2 weeks ago	117MB
ubuntu	12.04	18305429afa1	8 years ago	162MB

```
PS C:\Users\semen>
```

Рисунок 1 – Загрузка образов Ubuntu

Также проверим, имеются ли в системе контейнеры.

```
PS C:\Users\semen> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

```
PS C:\Users\semen>
```

Рисунок 2 – Запрос списка контейнеров

2 Изоляция

Посмотрим информацию о хостовой системе, выполнив команду hostname.

```
PS C:\Users\semen> hostname
ValeraMagistr
PS C:\Users\semen> hostname
ValeraMagistr
PS C:\Users\semen>
```

Рисунок 3 – Информация о системе

Вопрос: одинаковый ли результат получился при разных запусках?

Ответ: при запуске на одной машине, результат не будет меняться между запусками.

Попробуем выполнить то же самое в контейнерах.

```
PS C:\Users\semen> docker run ubuntu hostname
7a762732191d
PS C:\Users\semen> docker run ubuntu hostname
0e9722b76f93
PS C:\Users\semen>
```

Рисунок 4 – Информация о системе в контейнерах

Вопрос: одинаковый ли результат получился при разных запусках?

Ответ: при вводе этой команды, для каждой команды, создаются новые контейнеры, которые изолированы друг от друга, каждый контейнер имеет собственный идентификатор.

3 Работа с портами

Загрузим образ python.

```
Using default tag: latest
latest: Pulling from library/python
23b7d26ef1d2: Download complete
07d1b5af933d: Download complete
1eb98adba0eb: Download complete
e25cca11fd29: Download complete
171e1bee1949: Download complete
b617a119f8a2: Download complete
739b86d2a778: Download complete
Digest: sha256:34dc8eb488136014caf530ec03a3a2403473a92d67a01a26256c365b5b2fc0d4
Status: Downloaded newer image for python:latest
docker.io/library/python:latest
PS C:\Users\semen>
```

Рисунок 5 – Загрузка образа Python

Запустим встроенный в Python модуль веб-сервера из корня контейнера, чтобы отобразить содержание контейнера с портом 8000.

```
PS C:\Users\semen> docker run -it -p8000:8000 python python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Рисунок 6 – Запуск веб-сервера

Directory listing for /

- [.dockerenv](#)
- [bin/](#)
- [boot/](#)
- [dev/](#)
- [etc/](#)
- [home/](#)
- [lib/](#)
- [lib64/](#)
- [media/](#)
- [mnt/](#)
- [opt/](#)
- [proc/](#)
- [root/](#)
- [run/](#)
- [sbin/](#)
- [srv/](#)
- [sys/](#)
- [tmp/](#)
- [usr/](#)
- [var/](#)

Рисунок 7 – Содержимое корневой директории в контейнере

4 Именованные контейнеры, остановка и удаление

Запустим контейнер из прошлого раздела в фоновом режиме.

```
LaHaine@ValeraMagistr MINGW64 ~ (new_branch)
$ docker run -p8000:8000 --name pyserver -d python python -m http.server
917b78875b137e71f514658ce585b1aedd5f7df83bbb3fc19f5114181636d6a5

LaHaine@ValeraMagistr MINGW64 ~ (new_branch)
$ docker ps | grep pyserver
917b78875b13  python  "python -m http.serv..." 13 seconds ago Up 12 seconds 0.0.0.0:8000->8000/tcp pyserver
```

Рисунок 8 – Запуск контейнера и проверка

5 Постоянное хранение данных

Запустите контейнер, в котором веб-сервер будет отдавать содержимое директории /mnt.

```
LaHaine@ValeraMagistr MINGW64 ~ (new_branch)
$ docker run -p8000:8000 --name pyserver --rm -d python python -m http.server -d /mnt
280786f5803aaf88be89c316256205a36275fc1a32d46fc5792128e360ff57b4
```

Рисунок 9 – Запуск контейнера

Вопрос: что значат остальные флаги запуска? Где здесь команда, которая выполнится в контейнере?

Ответ: **-p8000:8000** выбор портов, левый – порт на хостовой машине, правый – порт внутри контейнера. **--name pyserver** – явное указание имени контейнера. **--name** – автоматическое удаление контейнера после завершения работы. **-d** – запуск контейнера в фоновом режиме.

Попадём в запущенный контейнер используя оболочку `bash` и создадим текстовый файл в `/mnt`.

```
LaHaine@ValeraMagistr MINGW64 ~ (new_branch)
$ docker exec -it pyserver bash
root@280786f5803a:/# cd mnt && echo "hello world" > hi.txt
root@280786f5803a:/mnt# exit
exit
```

Рисунок 10 – Создание файла внутри контейнера

Directory listing for /mnt/

-
- [hi.txt](#)
-

Рисунок 11 – Содержимое директории в контейнере

7 Dockerfile

Напишем `Dockerfile`, соберём образ и произведём запуск на хостовой машине.

```
FROM ubuntu:latest
RUN apt update \
&& apt install -y python3 fortune \
&& cd /usr/bin \
&& ln -s python3 python
RUN /usr/games/fortune > /mnt/greeting-while-building.txt
ADD ./data /mnt/data
EXPOSE 80
CMD ["python" , "-m" , "http.server" , "-d" , "/mnt/" , "80"]
```

Листинг 1 – Содержимое `Dockerfile`

```
PS C:\Users\semen\Desktop\MIREA\Теория разработки программных приложений\Практика4> docker build -t mycoolimage .
[+] Building 80.0s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 310B
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> CACHED [1/4] FROM docker.io/library/ubuntu:latest@sha256:1e622c5f073b4f6bfad6632f2616c7f59ef256e96fe78bf6a595d1dc4376
=> => resolve docker.io/library/ubuntu:latest@sha256:1e622c5f073b4f6bfad6632f2616c7f59ef256e96fe78bf6a595d1dc4376ac02
=> [internal] load build context
=> => transferring context: 26B
=> [auth] library/ubuntu:pull token for registry-1.docker.io
=> [2/4] RUN apt update && apt install -y python3 fortune && cd /usr/bin && ln -s python3 python
=> [3/4] RUN /usr/games/fortune > /mnt/greeting-while-building.txt
=> [4/4] ADD ./data /mnt/data
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:340b649832bd085da62ae085197b53a37ea2f1a9fa517600ab1216c4e11c1891
=> => exporting config sha256:a109cc6acdf2901d7d351dea7f9d87e7ac9d8d65dec8cba02a427ce91e92517d
=> => exporting attestation manifest sha256:4f73087a030cd121650b9b82cf096646b7c419b8dbc4f00b372b95b1c1f30311
```

Рисунок 12 – Сборка образа

```
PS C:\Users\semen\Desktop\MIREA\Теория разработки программных приложений\Практика4> docker run --rm -it -p8099:80 mycoolimage
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Рисунок 13 – Запуск

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Напишем Dockerfile, соберём образ и запустим контейнер согласно варианту.

Необходимо использовать базовый образ ubuntu:20.10, примонтировать файл data/student.txt как /mnt/files/student.txt в контейнере. Установить пакет: figlet.

```
FROM ubuntu:20.10

RUN sed -i 's/archive.ubuntu.com/old-releases.ubuntu.com/g' \
    /etc/apt/sources.list &&
RUN apt update && \
    apt install -y python3 figlet

EXPOSE 8802
CMD ["python3", "-m", "http.server", "-d", "/mnt/files", "8802"]
```

Листинг 2 – Содержимое Dockerfile

```
PS C:\Users\semen\Desktop\MIREA\Теория разработки программных приложений\Практика4> docker build -t golev .
[+] Building 61.5s (8/8) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile              0.1s
=> => transferring dockerfile: 459B                             0.0s
=> [internal] load metadata for docker.io/library/ubuntu:20.10 0.1s
=> [internal] load .dockerignore                                0.1s
=> => transferring context: 2B                                    0.0s
=> CACHED [1/4] FROM docker.io/library/ubuntu:20.10@sha256:a7b08558af07bcccc994b01e1c84f1d14a2156e0099fcf7fcf73f52d082791e 0.1s
=> => resolve docker.io/library/ubuntu:20.10@sha256:a7b08558af07bcccc994b01e1c84f1d14a2156e0099fcf7fcf73f52d082791e 0.1s
=> [2/4] RUN sed -i 's/archive.ubuntu.com/old-releases.ubuntu.com/g' /etc/apt/sources.list && sed -i 's/security.ubuntu.com/old-rele 0.4s
=> [3/4] RUN apt update && apt install -y python3 figlet && apt clean && rm -rf /var/lib/apt/lists/* 57.7s
=> [4/4] RUN mkdir -p /mnt/files                                0.5s
=> exporting to image                                           2.7s
=> => exporting layers                                           1.9s
=> => exporting manifest sha256:34bbcc846a8041161511d69a1299b61cb366697c3494c6826e01848fe4a96ff5 0.0s
=> => exporting config sha256:e6708b9e5af6d57a6043f48ad9386e233673fc61d7f560de4405715fcaaaae36 0.0s
=> => exporting attestation manifest sha256:4d83021b11671c336c1d0de00adee7e9301dbdbcab1addc91c6daa7d1026c8b6 0.1s
=> => exporting manifest list sha256:4435bd02c10b5b72438531f86074af54fba92eabc9593bcf60c715ac8ba5c1ff 0.0s
=> => naming to docker.io/library/golev:latest                  0.0s
=> => unpacking to docker.io/library/golev:latest               0.6s
```

Рисунок 14 – Сборка образа

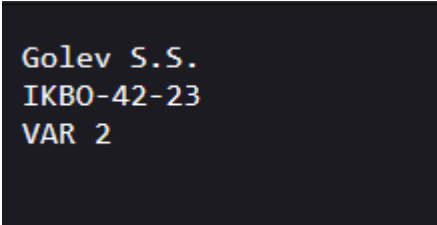
```
PS C:\Users\semen> docker run --rm -d -p 8802:8802 -v /c/Users/semen/data/student.txt:/mnt/files/student.txt golev
7747085f32630a1c53dfaa35dc4d3187fbdd761c7838ad5ba0168072fe4e8811
```

Рисунок 15 – Запуск контейнера

Directory listing for /

- [student.txt](#)
-

Рисунок 16 – Содержимое директории /mnt/files

A terminal window with a dark background and light-colored text. The text is displayed on three lines: 'Golev S.S.', 'IKBO-42-23', and 'VAR 2'.

```
Golev S.S.  
IKBO-42-23  
VAR 2
```

Рисунок 17 – Содержимое файла в директории /mnt/files

ВЫВОД

В ходе выполнения практической работы были изучены основные принципы работы с Docker и его компонентами.

Изучена работа с образами и контейнерами: загрузка образов, запуск контейнеров, просмотр списка контейнеров и образов. Были освоены команды `docker pull`, `docker run`, `docker ps` и другие.

Особое внимание уделено изоляции контейнеров, передаче переменных окружения, монтированию директорий и файлов с хоста в контейнер, а также постоянному хранению данных с помощью томов.

Собран собственный образ с использованием `Dockerfile`, в котором установлены необходимые пакеты и настроен запуск встроенного веб-сервера Python. Изучены принципы проброса портов и работы с внутренними и внешними портами контейнера.

В результате сформировано практическое понимание создания, настройки и управления окружениями на базе Docker.